Automatic Test Image Generation using Procedural Noise

Matthew Patrick, Matthew D. Castle, Richard O. J. H. Stutt and Christopher A. Gilligan
Department of Plant Sciences, University of Cambridge, United Kingdom
{mtp33, mdc31, rs481, cag1} @ cam.ac.uk

ABSTRACT

It is difficult to test programs that input images, due to the large number of (pixel) values that must be chosen and the complex ways these values interact. Typically, such programs are tested manually, using images that have known results. However, this is a laborious process and limited in the range of tests that can be applied. We introduce a new approach for testing programs that input images automatically, using procedural noise and spatial statistics to create inputs that are both realistic and can easily be tuned to have specific properties. The effectiveness of our approach is illustrated on an epidemiological simulation of a recently introduced tree pest in Great Britain: Oriental Chestnut Gall Wasp. Our approach produces images that match the real landscapes more closely than other techniques and can be used (alongside metamorphic relations) to detect smaller (artificially introduced) errors with greater accuracy.

CCS Concepts

•Software and its engineering → Software creation and management; Software testing and debugging;

Keywords

software testing; image processing; test data generation

1. INTRODUCTION

Images are used as program inputs in fields such as medical imaging [3], material analysis [9], computer vision [18] and urban/environmental modelling [16]. The outputs of this software are often used to make important (and sometimes even life-critical) decisions. It is therefore essential to test the software thoroughly, under a variety of conditions, to ensure it will work correctly when needed. For example, facial recognition failed to identify the suspects after the Boston Marathon bombing, even though their photographs were in an FBI database [2]. The problem was that the database photographs were taken under controlled conditions, whereas the images taken at the scene were of poor quality and from far away. We should therefore test image software with a large number of variations of each image.

Imaging software is often tested using an established data set, for which the expected results are known [18]. Examples include the Berkeley Segmentation data set [11] and the Caltech-256 object category data set [6]. Publicly available image data sets make it possible to compare the performance and accuracy of commonly used imaging techniques. However, suitable data sets are unlikely to exist for software that performs bespoke operations. It is labour intensive and can be error prone to create new data sets for each application, so automated test data generation would be useful.

Automated test data generation techniques can quickly produce a set of test data with specific properties [1]. However, few attempts have been made to address the challenges posed by image inputs. It is difficult to generate test images automatically because of their high dimensionality (large number of pixel values). The relationships between pixel values are also often more important than the individual pixels themselves. To address these challenges, we introduce a new automatic approach for generating test images that have specific target properties. Our approach can be used to generate variations of existing images, or to explore the input space of all possible images that could be used.

The approach introduced in this paper uses procedural noise and spatial statistics: two techniques from very different fields. Procedural noise has its origins in computer graphics for films and video games [13]. It allows complex, natural-looking images to be constructed using simple tunable parameters. By contrast, spatial statistics arose from a need to understand complex processes in geography and geology [16]. They are used to characterise, compare and make predictions about patterns in the spatial arrangements of values. Despite their differences, we have found these two techniques complement each other. Our approach combines procedural noise and spatial statistics in the following way:

1. Multiple nested layers of **procedural noise** generate realistic images, according to a set of parameters
2. The parameters are optimised with a **genetic algorithm**, using **spatial statistics** as the fitness function
3. **Metamorphic testing** is applied to compare the software output, using the optimised images as input

ASE '16 Singapore

ISBN 123-4567-24-567/08/06...$15.00

DOI: 10.475/123_4

We illustrate our approach on a specific example involving a highly complex epidemiological simulation, which inputs images representing the geographic distribution of host species and outputs a stochastic prediction of the epidemic progress over time. The results of this work represent a preliminary viability study of our approach. After showing our approach to be successful on this case study, we will continue our research by applying it to a wide variety of software, producing test images with a diverse range of properties.

2. LITERATURE REVIEW

There have been other attempts to generate images automatically for testing. Guderlei and Mayer [7] generated pixel values at random and positioned discs and squares randomly on the image. Just and Schweiggert [8] combine randomly produced red, green and blue layers into colour images.

It is also possible to modify existing images e.g. Koljonen and Alander [9] applied motion blur and Gaussian pixel noise to images of materials under stress and strain to test their optical extensometer still functioned correctly.

Mantere and Alander [10] used a genetic algorithm to create test images for digital halftoning that maximise the difference between the halftoned and original image. They found evolving pixel values to be slow (even on 16x16 images), so optimised the position, size and colour of shapes.

There are four main problems with the previous work:

1. **Precision:** [7] and [8] generate images randomly, so cannot be used to achieve specific properties
2. **Efficiency:** [10] found pixel value optimisation infeasibly slow, even with very small images
3. **Flexibility:** [9] adds noise to existing images; it cannot be used to test software with new images
4. **Scalability:** the largest image generated in the previous work was just 256x256 pixels [10].

3. OUR APPROACH

In contrast with previous research, our approach can be used to create more complex images with greater efficiency. The images produced by our approach are tuned to specific target properties, using spatial statistics as the fitness function to a genetic algorithm. This allows us to test **precise** features of the software by evolving multiple different images with the same properties. Rather than optimising the value of each pixel separately, we evolve simple parameters of procedural noise that have an effect across the entire image. This greatly simplifies the search space, to make optimisation more **efficient**. Our approach is also highly **flexible**, able to automatically tailor the images generated to be similar to those used by the particular software under test. It is **scalable** to larger images than those in the previous studies (the images generated in this paper are 389x252 pixels). The following subsections outline the main components of our approach (procedural noise, genetic algorithms, spatial statistics and metamorphic testing) and Algorithm 1 summarises the steps involved in generating test image data.

Algorithm 1 Automated test image generation

```
Choose target Moran's I and pixel value histogram
Initialise candidates (random number of layers [each
  composed of 8 sub-layers], frequencies and amplitudes)
while stopping condition not met do
    Generate images from candidates using Perlin noise
    Transform images to match pixel value histogram
    Calculate Moran's I values on generated images
    Select fittest images (smallest difference in Moran's I)
    Apply mutation and crossover to update population
end while
Utilise the fittest images in metamorphic testing
```

3.1 Procedural Noise

The most basic form of procedural noise is white noise [14]. White noise is produced by sampling individual pixel values independently, using a pseudo-random number generator. Although its probability distribution may be adjusted, white noise does not take into account the relationships between neighbouring values. Images therefore appear patternless and cannot immediately be used to describe natural processes. Nevertheless, more realistic images can be produced by combining and transforming white noise in various ways. These transformation functions are used to create multiple complex sequences of values, that may be reproduced using a set of simple parameter values (and a random seed). This is helpful for our purposes, as we do not need to store every image we evaluate - just the parameters that were used.

Perlin noise [13] is a well known procedural noise function that generates values by combining random vectors produced (using white noise) at each vertex of an interpolation grid. Unlike white noise, the output of Perlin noise changes smoothly from one value to the next. Perlin noise is parametrised by its frequency (grid resolution) and amplitude (the size of contribution it makes to the image). Layers of Perlin noise (produced using different parameters) can be added together to create more complex and natural-looking images [3]. The frequencies and amplitudes of each layer are attenuated, to create a fractal-like effect. In our approach, we combine multiple sets of attenuated Perlin noise layers, each with a different initial frequency and amplitude. This allows for heterogeneity to occur at multiple levels of detail and for some layers to be made more prominent than others.

3.2 Genetic Algorithms

We decided to tune the parameters of our image generation technique using a genetic algorithm because procedural noise generation is stochastic and we found the fitness landscape to be noisy. Under these conditions, local search techniques such as hill climbing can become stuck in local optima. Genetic algorithms avoid this problem by balancing the exploitation of existing solutions and exploration of new values, using variation and selection [4]. Variation creates new candidates by mutating and recombining the existing solutions. The fittest candidates are selected and the weaker candidates removed, so as optimisation progresses, there is a trend towards increasingly fitter solutions. In our approach, candidate solutions are structured as variable length lists, whereby each item in the list represents a layer of noise (with a specific initial frequency, amplitude and random seed). This information is sufficient to reproduce the image, without needing to store the individual pixel values of each candidate considered by the genetic algorithm.

3.3 Spatial Statistics

Spatial statistics are used to characterise real world processes from a variety of fields, including economics, criminology, urban/environmental studies and epidemiology [16].

First order metrics, such as population density, measure the composition of a landscape (i.e. how much of each property the landscape has in each area). By contrast, second order metrics assess the landscape's configuration (i.e. whether there are spatial patterns in the data). Our approach makes use of both first and second order spatial statistics.

We use a first order statistic (histogram of pixel values) to ensure images are generated according to a target distribution. The images produced by our technique have their pixel values scaled, such that the desired histogram distribution is achieved. This is done before the Moran's I fitness score calculation, to ensure this step does not affect the second order spatial properties of the resulting images.

Moran's I [12] measures autocorrelation by comparing the values that occur in a particular neighbourhood with the average value for the landscape. It is based on Pearson's Correlation Coefficient (see Equation 1) and measures how values change over different spatial lags (i.e. distances between pixels). A positively auto-correlated landscape, which has all the high values clustered into one region and all the low values in another, will have a Moran's I value around 1. By contrast, a negatively auto-correlated landscape, which has an alternating pattern of high and low values, will have a Moran's I value around -1. A landscape with complete spatial randomness will have a Moran's I value of 0.

$$I = \frac{N}{\sum_i \sum_j w_{i,j}} \frac{\sum_i \sum_j w_{i,j}(X_i - \bar{X})(X_j - \bar{X})}{\sum_i (X_i - \bar{X})^2} \quad (1)$$

($w_{i,j}$ is a matrix of neighbourhood weights between i and j)

3.4 Metamorphic Testing

For many complex programs (such as the epidemiological simulation in our case study) it is difficult to know what the correct outputs of the software should be, without using the software itself to find out the answer. It is therefore just as challenging to construct a test oracle as it is to develop the software correctly in the first place. Metamorphic testing [15] addresses this problem by circumventing the need to calculate the expected output values directly. Instead, they use information about how the output is expected to change when the inputs are adjusted in a particular way.

In this paper, we illustrate how the images produced by our approach can be applied to metamorphic testing of an epidemiological simulation. We evolve images that have the same spatial properties as the original landscape and compare the outputs of the simulation on each image. Since the spatial properties are the same, epidemics should progress at a similar rate. However, our simulation is stochastic, so it produces a different result each time it is used. We therefore compare the distributions of areas under the epidemic progress curves using a Kolmogorov-Smirnov test. If the distributions are different, there may be a fault in the software.

4. IMPLEMENTATION DETAILS

We use a form of Perlin noise known as ridged turbulence ($f(x) = 1 - |\text{perlin}(x)|$). Each (dynamically evolved) layer is composed of eight sub-layers, with persistence (0.5) and lacunarity (2.0). Initially, each candidate is given a Poisson random number of layers (with rate 0.1). The amplitude for each layer is set to a uniform random number between 0 and 10 and the initial frequency is set according to the equation $\exp(0.5i)$, where $i \in [0 \dots N]$ is the index of each layer.

Our genetic algorithm uses Gaussian mutation ($\sigma^2 = 1$), single-point recombination and roulette wheel selection. The probability of mutation is 0.5 and the probability of recombination is 0.9. We use a population size of 20 and a fixed number of generations (50). Candidate images have their pixel value scaled using a log-log histogram with 10 000 bins. Moran's I is measured over 50 spatial lags using the Euclidean distance between pixels. The calculated values are compared (for relative difference) with the target values for the image, using the sum of squares difference metric.

5. CASE STUDY

We illustrate our approach on a simulation of Oriental Chestnut Gall Wasp (OCGW) [5]. An image is input representing the density of hosts in each grid cell (plus initial conditions), then the simulator outputs a set of curves predicting how the pest is likely to spread. It is important the results are correct, as they are used to advise government policy decisions for control. However, OCGW was first observed in Great Britain in 2015 [5] and so far has only been found at a few sites in South-East England. This lack of data makes it difficult to know what the correct outputs should be, so metamorphic relations are important for testing. We use the relation that landscapes which have the same spatial properties should produce the same epidemiological results. In particular, we are interested in knowing whether the β parameter (rate of secondary infection) is processed and interpreted correctly. This is important as it affects how quickly an infected host infects the hosts that surround it.

6. RESEARCH QUESTIONS

RQ1: Do the images produced by our technique match the target properties more closely than those produced by alternative techniques?
This question is evaluated by setting the target properties to be the same as the original image (using the Moran's I statistic and pixel value histogram). We compare our technique with two alternatives (random image generation [7] and the Poisson Cluster Process (PCP) [17]) to generate images that recreate these properties. The alternative approaches are often used in spatial ecology research. If our technique is successful, the progress of the epidemic on the original image should be more similar to that on the images produced by our approach than the other two techniques.

RQ2: Is our technique more efficient at creating suitable images than the alternative techniques?
In addition to being able to create test images with the target properties, it is also important our technique is efficient. If it takes too much time, then it may be difficult to use our technique in practice. We therefore compare the rate at which images produced by our technique approach the target properties (i.e. lower the fitness score) and the time that is taken for these images to be produced. If our approach is more efficient, it should be able to achieve the target properties more quickly than the alternative techniques.

RQ3: Can our technique be used to find faults more effectively than the alternative techniques?
Finally, we consider the fault-finding effectiveness of

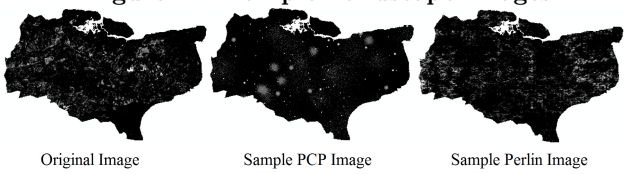
the images produced by our technique, using metamorphic relations. Artificial faults are introduced into the β parameter of the simulation (the rate of secondary infection). As the value of β is increased, we expect a difference to be detected more often (i.e. fewer false negatives). However, when β is left unchanged, no difference should be detected most of the time (few false positives). In this way, we can compare our technique with the alternative approaches in terms of the number of false negatives and positives it produces.

7. EXPERIMENTAL METHODOLOGY

Our original image represents the Sweet Chestnut landscape in Kent, where OCGW was first found in Great Britain (see Figure 1), produced using data from the Forestry Commission Sub-Compartment Database and National Forest Inventory. The landscape resolution is set to 250m, which makes the image 389 x 252 pixels. We evaluate the epidemiological similarity of the new images with the existing image by running 100 simulations from different starting locations on each of 100 images produced by the 3 techniques: Random, PCP and Perlin (see Figure 1).

Random images are produced with a single layer of white noise (we select the fittest solutions found using the same number of evaluations as the PCP and Perlin noise approaches), but PCP was extended using our multi-layer evolutionary approach. Each layer consists of 10 clusters and each cluster contains 1000 points. Points are normally distributed, with the centre of each cluster following a Poisson distribution. Amplitude is interpreted as with our (Perlin noise) technique, but instead of frequency, we optimise the variance of the cluster distribution. Our PCP approach is therefore a novel technique, but it was necessary to augment it this way to provide a competitive alternative.

Figure 1: Example Landscape Images



8. RESULTS

8.1 Answer to RQ1

There are two ways in which we can compare the images produced by the techniques under evaluation with the original image of the Sweet Chestnut landscape. First, we should consider spatial statistics. Since Moran's I is used in the fitness function for our genetic algorithm, these differences provide information as to how well the optimisation process has achieved our target spatial statistics (i.e. similarity with the original image). We can then evaluate how similar the outputs of the simulation are for each image, in terms of the rate at which OCGW spreads. This tells us how effective optimising the spatial statistics has been in achieving epidemiological similarity with the original image.

Table 1 summarises the differences in spatial statistics and epidemiological properties between the original image and the images produced by each technique: random image generation, the Poisson Cluster Process and Perlin noise (proposed by this paper). Spatial differences are measured using the sum square error in Moran's I. Unsurprisingly, the differences are far greater for the random technique than for PCP or Perlin. This may be because random image generation is not optimised using the genetic algorithm applied by the other two techniques. However, the difference is 3 times larger for the Poisson Cluster Process than for our Perlin approach. It is therefore not just the genetic algorithm that makes the difference, the image generation technique is also important. Perlin noise produces more realistic images and makes it easier to achieve spatial similarity by providing parameter values that have an effect across the entire image.

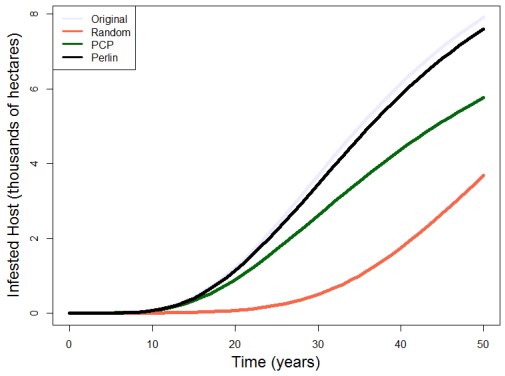
Figure 2 shows the mean progress curves of OCGW in simulations run using the images produced by each technique. Although the simulations had not yet reached a point of equilibrium, we stopped them after 50 years, as predictions made further into the future seem over-reaching. The differences between the techniques are very clear. Epidemics take off very slowly when the images are produced randomly. Some of the Poisson Cluster Process images offer realistic results, but the variance is much higher. Images produced using our Perlin approach are by far the most representative in terms of the similarity of their epidemic progress curve.

The areas under the epidemic progress curves (AUC) confirm these results numerically (see Table 1). The mean difference in AUC for Perlin noise is over 5 times smaller than PCP and 14 times smaller than the randomly produced images. It is also significant that the AUC results match the Moran's I results, when ordered by size of difference. In both cases, Perlin noise images are most similar to the original, whereas randomly produced images are the least similar (they also have the smallest variance, since they are spatially homogeneous). This suggests Moran's I is a suitable metric for optimisation, since images with the smallest differences in Moran's I are also the most similar epidemiologically.

Table 1: Spatial and Epidemiological Differences

	Δ Moran's I		Δ AUC	
	Mean	SD	Mean	SD
Random	50.0	0.071	113	3.17
PCP	6.61	1.92	43.6	20.7
Perlin	1.86	0.782	7.97	12.9

Figure 2: OCGW Epidemic Progress Curve



8.2 Answer to RQ2

We evaluate RQ2 by investigating the amount of time required by each technique to generate test images and the rate at which the fitness of the images improves. Table 2 shows

random image generation is the fastest technique (taking 67.2 minutes on average), whereas the Poisson Cluster Process is the slowest (105 minutes on average). Despite the large variance for PCP, all these differences are statistically significant: Student's t tests (with Bonferroni correction) give p-values of 2.43×10^{-17}, 0 and 6.81×10^{-6} (for the differences between random and PCP, random and Perlin, PCP and Perlin respectively), with Cohen's d effect sizes of 1.19, 9.43 and 0.615 respectively. Random image generation is therefore the fastest of the three techniques.

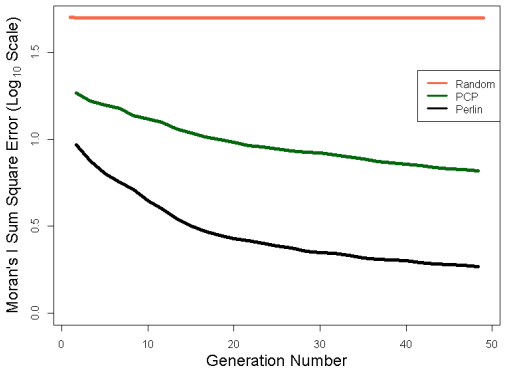
Figure 3 shows how the Moran's I sum square error improves as more candidates are evaluated. Random image generation only reduces the error by 0.568% of its initial value, which makes it inefficient despite its advantages in speed. By contrast, PCP and Perlin noise substantially reduce their error as optimisation progresses (by 64.2% for PCP and 79.9% for Perlin noise). Their fitness had not quite stabilised after 50 generations, so this is likely to have improved further, but we need to be aware most testers will only be willing to wait for a certain amount of time.

Our Perlin noise approach is more efficient than PCP, as it reduces the error by a greater amount over a fixed number of generations and each generation takes less time. PCP may be slower because it adds 3 times more layers than Perlin noise as it struggles to match the target properties (see Table 2). Further improvements might be possible by simplifying the spatial statistic (e.g. reducing the number of lags) or procedural noise (e.g. reducing the number of layers) but we need to trade this against accuracy in the result.

Table 2: Execution Time and Number of Layers

	Time (min)		Layers	
	Mean	SD	Mean	SD
Random	67.2	0.858	1.00	N/A
PCP	105	45.0	42.4	36.1
Perlin	85.4	2.59	18.3	10.1

Figure 3: Optimisation Progress



8.3 Answer to RQ3

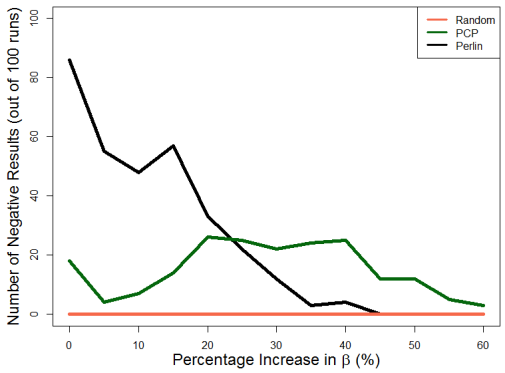
Finally, RQ3 is evaluated by introducing artificial faults into the β input parameter of the simulations run on the generated images. We then count the number of negative results of Kolmogorov-Smirnov tests, comparing the distribution of areas under the epidemic progress curves on newly produced images with those of the original image. We consider a result to be negative if the p-value is greater than 0.05. It is necessary to compare the distributions of values in this way, since the simulation we are testing is stochastic. However, there is still a possibility some p-values will be greater than 0.05 due to random chance. We address this problem by running 100 simulations on 100 different images, starting each one from a different location, and then counting the number of negative results out of the 100 images.

Figure 4 shows the results of these experiments. We can interpret them in terms of the numbers of false positives and negatives. A false positive occurs when a test indicates a fault that does not exist. To see these, we need to look at the results where $\beta = 0\%$ (i.e. no change has been made). The number of false positives is 14 out of 100 for our Perlin noise approach, but 82 for PCP and 100 for the random image generation. Unless Perlin noise is used, we are likely to conclude the simulation is faulty when it is correct.

The number of false negatives can be seen when β is increased. Random image generation produces no false negatives, but this is misleading since its epidemic progress curve is too dissimilar to produce any negative results. PCP has its highest number of negative results when β is increased by 20%, i.e. it has more false negatives than true negatives. The highest number of negative results should occur when $\beta = 0\%$, but the PCP epidemic progress curve lies below that of the original image (see Figure 2), so artificial faults that increase β move it closer to the correct position.

Our Perlin noise approach is the most useful of the three techniques, since it has the lowest number of false positives and its general trend is to reduce the number of false negatives as the size of the fault gets larger (i.e. as β increases). There are a couple of points at which the number of false negatives increases when it should be decreasing, but this is likely to be due to stochasticity. However, it only reaches zero false negatives once β is increased by 45% (depending on the application, this could be considered a large fault). 20% increases in β may be detected by considering cases in which there are more than 50 positives as a possible fault. We could also increase the number of runs and lower this cut-off further (but this is likely to be more expensive).

Figure 4: OCGW Negative Results



9. THREATS TO VALIDITY

We were careful to address any internal threats to validity by repeating our experiments over 100 trials on 100 images produced using each technique. Since we are comparing three different approaches (Random image generation, Pois-

son Cluster Process and Perlin noise), we need to be aware of the multiple comparisons problem. Therefore, when using the Student's t-test, we applied the Bonferroni correction.

External threats to validity are potentially more significant to our work. We have shown our approach to generate test images more effectively than the alternatives, but there is no guarantee it will work as well for other software. We plan to evaluate this issue in our future work. However, the approach we have presented is modularised, so that it is easy to adjust each component to further tailor its performance.

10. CONCLUSIONS

Programs that input images are difficult to test because of the large input domain of pixel values that need to be explored. We have introduced a new approach to address this problem using procedural noise, spatial statistics, genetic algorithms and metamorphic testing. Our approach can be applied to generate complex, natural images that are tuned to specific properties for use in testing. The image generation process is controlled by a set of simple parameters that can be efficiently evolved by a genetic algorithm.

We illustrated our approach on a stochastic spatially-explicit simulation for a tree pest in Great Britain. This software is difficult to test because of its complexity and the lack of a suitable oracle or test data. However, we showed that in situations such as this, we can use our technique to produce new image test data with similar spatial properties as the original image. By applying metamorphic relations, this test data can be used to identify faults more accurately, with fewer false positives than the alternative techniques.

11. FURTHER WORK

We intend to continue this work by applying our approach to software in a variety of different fields, such as medical imaging, material analysis and computer vision. We will also use the images our technique produces to test software in new ways e.g. through other metamorphic relations. The aim of future work will be to discover the capabilities of our approach, identifying the types of software and tests it is most effective for. Throughout this process, we will evaluate the various implementation decisions we have made using sensitivity analyses on each application, so as to tune our approach to be as effective as possible in every situation.

We will extend our approach for use with images that have a more complex structure, such as those which involving several distinct parts. One way to achieve this is by combining our approach with the shape overlaying technique of other researchers [7][10]. Each shape can contain a different 'texture', produced by our approach. We can use our approach to generate images with a wide variety of target properties, not just those that match the original images. We will therefore investigate how best to explore these options, considering techniques such as fuzz testing and program analysis [1] to generate suitable test images as efficiently as possible.

12. ACKNOWLEDGMENTS

This work was supported by the University of Cambridge/ Wellcome Trust Junior Interdisciplinary Fellowship "Making scientific software easier to understand, test and communicate through modern advances in software engineering".

13. REFERENCES

[1] S. Anand, E. Burke, T. Chen, J. Clark, M. Cohen, W. Grieskamp, M. Harman, M. J. Harrold, and P. McMinn. An Orchestrated Survey of Methodologies for Automated Software Test Case Generation. *J. Systems Software*, 86(8):1978–2001, 2013.

[2] Andrew Leonard. Why facial recognition failed. http://www.salon.com/2013/04/22/why_facial_recognition_failed/, 2013. Accessed: 2016-04-26.

[3] M. Dustler, P. Bakic, H. Petersson, P. Timberg, A. Tingberg, and S. Zackrisson. Application of the Fractal Perlin Noise Algorithm for the Generation of Simulated Breast Tissue. *SPIE Medical Imaging*, 9412, 2015.

[4] A. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Springer, Heidelberg, Germany, 2nd edition, 2015.

[5] Forestry Commission. Oriental Chestnut Gall Wasp. http://www.forestry.gov.uk/forestry/beeh-9xjbhf, 2016. Accessed: 2016-04-24.

[6] G. Griffin, A. Holub, and P. Perona. Caltech-256 Object Category Data Set. Technical Report 7694, California Inst. of Technology, 2007.

[7] R. Guderlei and J. Mayer. Towards Automatic Testing of Imaging Software by Means of Random and Metamorphic Testing. *Int. J. Software Engineering Knowledge Engineering*, 17(6):757–781, 2007.

[8] R. Just and F. Schweiggert. Evaluating Testing Strategies for Imaging Software by Means of Mutation Analysis. In *Proc. 2nd IEEE Int. Conf. Software Testing, Verification and Validation Works.*, pages 205–209, 2009.

[9] J. Koljonen and J. T. Alander. Deformation Image Generation for Testing a Strain Measurement Algorithm. *SPIE Machine Vision Pattern Recognition*, 47(10), 2008.

[10] T. J. Mantere and J. T. Alander. Automatic Image Generation by Genetic Algorithms for Testing Halftoning Methods. *Intelligent Robots and Computer Vision*, 4197, 2000.

[11] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *Proc. 8th IEEE Int. Conf. Computer Vision*, pages 416–423, 2001.

[12] P. Moran. Notes on Continuous Stochastic Phenomena. *Biometrika*, 37(1):17–23, 1950.

[13] K. Perlin. An Image Synthesizer. *Computer Graphics*, 19(3):287–296, 1985.

[14] M. Petrou and C. Petrou. *Image Processing: The Fundamentals*. Wiley, Chichester, UK, 2010.

[15] S. Segura, G. Fraser, A. B. Sánchez, and A. Ruiz-Cortés. A Survey on Metamorphic Testing. *IEEE Tran. Software Engineering*, 2016 [in press].

[16] J. Stilwell and G. Clarke. *Applied GIS and Spatial Analysis*. Wiley, Chichester, UK, 2004.

[17] R. L. Streit. *Poisson Point Processes: Imaging, Tracking, and Sensing*. Springer, New York, NY, 2010.

[18] R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, London, UK, 2010.